云原生应用的构建之路

八大步骤带领您完成构建

电子书

目录

1. 速度：数字化业务的必备要素 3
2. 什么是云原生应用？ 3
3. 传统应用与云原生应用 4
4. 云原生应用开发和部署的四大原则 6
5. 云原生应用的构建之路：八大步骤 7
   - 步骤 1：发展 DevOps 文化和实践 7
   - 步骤 2：借助高速单体式技术，为现有应用提速 7
   - 步骤 3：借助应用服务，加快开发速度 8
   - 步骤 4：欲善其事，必先择其利器 8
   - 步骤 5：构建可按需提供的自助式基础架构 9
   - 步骤 6：实现 IT 自动化，加速应用交付 9
   - 步骤 7：实施持续交付和高级部署技术 10
   - 步骤 8：推动变革，采用模块化程度更高的架构 11
6. 云原生应用业务案例 12

“对于提高市场份额，数字化驱动型企业的成功机率是其他企业的 8 倍；然而，它们仍然没有达到所谓的原生数字化。”

Bain 调查：传统企业的数字化进程和容器的重要作用

1. 速度：数字化业务的必备要素

随着数字化业务的不断推进，各种思维创新型技术应运而生：移动设备、智能传感器、可穿戴设备、虚拟现实、聊天机器人、区块链、机器学习和其他技术。在某种程度上，这也反映出了全新数字化原生型业务的迅猛发展势头。数字化业务不仅颠覆了传统的业务模式，还撼动了已有的企业和行业格局。对大多数企业而言，数字化业务意味着大刀阔斧推行企业敏捷性文化——只有借助更加迅速、更加灵活的开发和交付模式，才能满足各种快速变化的需求。由于大部分企业都无力承受全面重建技术基础，或即刻采用全新的业务实践和理念，因此，他们正从文化、流程和技术层面出发，逐步实现根本性转变，以求提升速度和敏捷性。

软件的影响力正在日益凸显，它不但会影响用户与企业间的互动方式，还会影响企业为保持市场竞争力而做出的创新之举。因此，应用的快速开发和交付已成为数字化企业必须要满足的一项新需求。

云原生方案会基于云技术原理，利用相关服务，并采用为实现云计算敏捷性和自动化而过优化的各种流程，从而**完成现有应用的现代化转型并构建新的应用**。本电子书会详细介绍各个相关步骤，帮助您根据自身现状，顺利地采用云原生应用方案。

*“‘云原生’是一个用来描述应用、架构、平台/基础架构和流程的形容词。只要能全面地考虑到这些要素，我们就能够以一种**经济高效**的工作方式来**提升我们的能力，以便快速应对各种变化**，并**降低不可预测性**。”*

CHRISTIAN POSTA
红帽首席架构师、
《面向 JAVA 开发人员的微服务》 作者

来源：INFOQ，“DEFINING CLOUD NATIVE: A PANEL DISCUSSION”（定义云原生：专题讨论），2017 年。

2. 什么是云原生应用?

云原生应用旨在充分利用云计算模型，从而提高速度、灵活性和质量并降低部署风险。虽然名字中包含“云原生”三字，但云原生方案的重点并不是应用部署在***何处***，而是***如何***构建、部署和管理应用。

云原生方案与微服务架构类似。然而，尽管微服务可通过构建云原生应用来交付，可企业仍需采取许多措施，才能在生产中熟练地管理微服务。而要想享受云原生应用的各种益处，也并非一定需要微服务。很多企业都通过基于相同的原则，构建出更优秀的模块化单体式应用，从而取得云原生方案的种种效益。

向云原生应用的开发和交付转型，是一次全方位的变革，涉及企业的文化、流程、架构和技术。因此，这是企业的必经之路，而不是所要到达的目标。这段旅程代表着一个变革周期，而实现变革从来不是一件容易的事。

3. 传统应用与云原生应用

云原生应用开发与传统应用开发之间的差别，充分彰显了变革的必要性。

表1. 传统应用开发与云原生应用开发

	传统	云原生
重点关注	使用寿命和稳定性	上市速度
开发方法	瀑布式半敏捷型开发	敏捷开发、DevOps
团队	相互独立的开发、运维、质量保证和安全团队	协作式 DevOps 团队
交付周期	长	短且持续
应用架构	紧密耦合 单体式	松散耦合 基于服务 基于应用编程接口（API）的通信
基础架构	以服务器为中心 适用于企业内部 依赖于基础架构 纵向扩展 针对峰值容量预先进行置备	以容器为中心 适用于企业内部和云环境 可跨基础架构进行移植 横向扩展 按需提供容量

3.1 传统应用的开发和交付

业务运营所需的很多基础性应用在设计时都未曾考虑提供数字化体验。它们拥有较长的生命周期，并被构建成了紧密耦合的单体式应用。它们符合定义明确的相关规范，但是这些规范的制定时间往往远早于应用的交付时间。

这些开发方案大部分都属于瀑布式和渐进式，不仅时间跨度长，而且直到近期才实现了半敏捷性。应用历经开发、测试、安全合规监管、部署和管理阶段，这些阶段被分隔成了不同的功能领域，每个领域都由不同的团队负责、发挥着不同的作用、肩负着不同的职责，且各方间均通过线性流程来沟通。

这些应用被构建成了紧密耦合的大型多功能应用。无论技术环境如何，用户界面、各种应用服务、数据访问代码和其他组件均被整合到了单个应用中。例如，如果某个电子商务应用被构建成了紧密耦合的单体式应用，那么这个应用通常会涵盖网购所需的所有功能，即 Web 用户界面、产品目录、购物车、产品推荐、产品评分和评论、支付系统以及其他组件——所有这些功能都在一个应用中。

对于大多数传统应用来说，基础架构会针对应用所需的峰值容量预先进行置备；还会通过纵向扩展来提高服务器的硬件容量，以实现容量扩展。

到 2020 年，在从私有数据中心迁移至公共云的模式 1* 应用中，超过 50% 的应用都将按照云原生架构规则来重写；而在 2017 年，这一比例还不足 10%。

Gartner：“Why You Must Begin Delivering Cloud-Native Offerings Today, Not Tomorrow”（为什么从今天就要开始交付云原生产品？），2018 年 1 月

3.2 云原生应用的开发和交付

由于云原生应用非常注重上市速度，因此，在开发时需要实施更加敏捷且基于服务和 API 的方案和持续交付策略。而这些方案能否成功实施又取决于以下几点：开发和交付团队间的 DevOps 协作；模块化程度更高的架构；能够按需横向扩展、支持多种环境并实现应用可移植性的灵活基础架构。

在利用现代化云技术实现灵活性和敏捷性后，企业会想将传统应用迁移到云环境中，以便进一步提高敏捷性，并充分利用按需提供的计算容量。

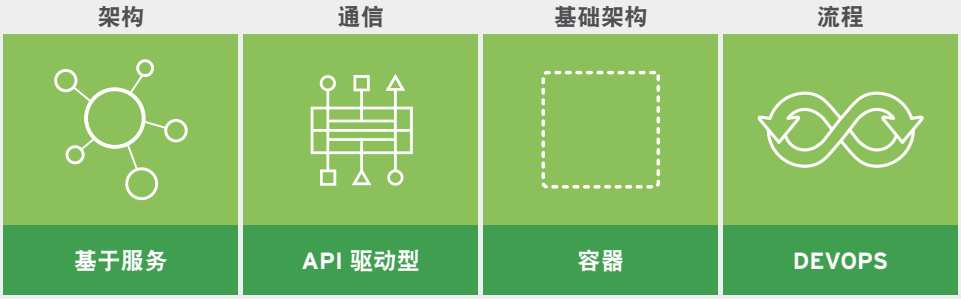
但是，传统平台中内置的很多运维功能，要么对于云环境而言早已过时，要么云环境本身就会提供并实施这些功能。云环境不但能简化主机的生命周期管理，还能帮助企业充分利用不变的基础架构原则，并对主机进行精调以满足单个应用实例的需求。

每个企业的云原生应用的构建之路可能都不尽相同。仅凭创建微服务无法实现数字化业务所需的服务质量和交付频率。同样地，仅凭采用支持敏捷开发或 IT 自动化的工具也无法加快云原生方案的速度。要想取得成功，企业必须综合考虑相关的实践、技术、流程和理念。

** **Gartner 将双模式定义为**对两种既相互独立、又相互关联的工作方式进行管理的做法：一种方式注重可预测性，另一种则注重探索。**模式 1 针对更易预测和理解的领域进行了优化。这种模式会着重利用已知内容，并革新传统环境以使其符合数字化时代的需求。**模式 2 则会不断探索，以尝试解决各种新问题并针对不确定的领域进行优化。*

4. 云原生应用开发和部署的四大原则

云原生应用开发所构建和运行的应用，旨在充分利用基于四大原则的云计算模型：基于服务的架构、基于 API 的通信、基于容器的基础架构以及 DevOps 流程。



基于服务的架构	基于服务的架构（如微服务）提倡构建松散耦合的模块化服务。其他模块化架构方案（如迷你服务）采用基于服务的松散耦合设计，可帮助企业提高应用创建速度，但不会增加复杂性。
基于 API 的通信	服务通过与技术无关的轻量级 API 来提供，这些 API 可以降低与部署、可扩展性和维护相关的复杂性和费用。企业可以通过所提供的 API 在内部和外部创建新的功能，创造新的商机。 采用基于 API 的设计时，只能在网上通过调用服务接口来通信，这样可避免因直接链接、共享内存模型或直接读取其他团队的数据存储而带来的风险。这种设计能让应用和服务延伸到不同的设备、拓展出不同的形式。
基于容器的基础架构	云原生应用依靠容器来构建跨技术环境的通用运行模型，并在不同的环境和基础架构（包括公共、私有和混合云）间实现真正的应用可移植性。容器技术会利用操作系统虚拟化功能来划分多个应用的可用计算资源，并确保这些应用安全无虞、相互独立。 云原生应用采用横向扩展模式，因此，只需添加更多应用实例，即可增加容量，而且这样的添加常通过容器基础架构中的自动化功能来实现。 由于容器的费用低、密度高，因而可在同一虚拟机或物理服务器内托管大量容器，这使得容器成了交付云原生应用的理想之选。
DEVOPS 流程	采用云原生方案时，企业会使用敏捷的方法，依据持续交付和 DevOps 原则来开发应用，这些方法和原则要求开发、质量保证、安全、IT 运维团队以及交付过程中所涉及的其他团队以协作方式构建和交付应用。

半数以上 (51%) 的大型企业都已实施 DevOps。但是，其中的大部分企业目前只在开发 10-40%（通常为 20%）的应用时使用了 DevOps 方案。

用于开展开发人员调查的 IDC PaaS 视图，2017 年 11 月

5. 云原生应用的构建之路：八大步骤

步骤 1：发展 DEVOPS 文化和实践

要完成云原生应用的构建之路，开发和 IT 运维团队必须进行多方面的变革，以便更加快速高效地构建和部署应用。无论身处哪个行业、规模如何，所有企业都需要周全地考虑各种活动、技术、团队和流程，因为这些要素综合起来才能实现 DevOps 文化。要想充分利用新技术，采用更加快速的方案，实现更为密切的合作，企业必须切实遵循 DevOps 的原则和文化价值，并围绕这些价值来进行组织和规划。

身处日新月异的数字化创新时代，管理多个分布式环境、高度定制的传统应用以及各种全新的应用工作负载会十分复杂，DevOps 对于一些企业而言可能会充满挑战。对于在应用组合中拓宽 DevOps 的实用范围，企业仍有很多潜能有待发掘。

DevOps 文化的推行不仅要靠工具和技术，也取决于员工是否愿意和信任用集成度和协作度更高的方案来开发和交付应用。企业可以把开源软件项目的文化作为参考，来构建 DevOps 文化。

红帽开放创新实验室可以引导企业完成 DevOps 之旅，倡导大胆尝试、迅速纠误、提升决策透明度，并通过表扬和奖励来促进团队间的相互信任和紧密协作。在专为实现创新而打造的环境中，团队将利用创新开源技术来快速构建原型，实现 DevOps 和敏捷工作流程。

进一步了解红帽开放创新实验室如何帮助您完成 DevOps 之旅
下载电子书

“如果您都不能构建一个结构合理的单体式应用，那又凭什么认为自己能建构一组结构合理的微服务？”

SIMON BROWN
CODINGTHEARCHITECTURE.COM/PRESENTATIONS/SA2015-MODULAR-MONOLITHS

步骤 2：借助高速单体式技术，为现有应用提速

在开启云原生应用之旅时，企业不能只关注开发新的应用。很多传统应用都是确保企业顺利运营和不断创收的关键所在，不能简单地取而代之。企业需将这类应用与新的云原生应用整合到一起。但是，您该如何加快现有单体式应用的运行速度？正确的方法是：将您现有的单体式架构迁移到模块化程度更高、且基于服务的架构中，并采用基于 API 的通信方式，从而实施快速单体式方案。

在开始实施将单体式应用重构为微服务的艰巨任务前，企业应该先为他们的单体式架构奠定坚实的基础。虽然单体式应用敏捷性欠佳，但其之所以受到诟病主要是因为自身的构建方式。然而，运行快速的单体式应用可以实现微服务所能带来的诸多敏捷性优势，而且不会增加相关的复杂性和成本。

通过对快速单体式方案进行评估，可以确保应用在构建时遵循了严苛的设计原则，以及正确定义了域边界。这样，企业就能在需要时以更加循序渐进、风险更低的方式过渡至微服务架构。如能以这种方式实现快速单体式应用的转型，即可为优良的微服务架构打下扎实的基础。

如果应用并非采用快速单体式方案而设计，您仍可将现有单体式应用迁移至基于容器的平台，以此提高这些应用的运行速度。这一转变不但能加速部署，还能提高投资回报（ROI）。通过云原生技术和方案，您将能实现单体式应用的后续整合或功能开发。

您还可以按照自己的步调分阶段将单体式应用划分成更小的组件。

步骤 3：借助应用服务，加快开发速度

可复用性一直都是加速软件开发的关键所在；云原生应用也不例外。但是，云原生应用的可复用组件必须经过优化，并整合到底层云原生基础架构中，以便充分发挥复用优势。

既然可以使用经过优化并整合到基于容器的底层基础架构中的现有应用，那为什么还要重建缓存服务、规则或工作流引擎、整合连接器、移动和 API 管理功能、数据虚拟化服务、消息代理或无服务器框架呢？无论是软件即服务（SaaS）、平台即服务（PaaS）还是 iPaaS 产品，这些应用服务都是非常高效的即用型开发人员工具。

云原生应用可能需要一种或多种此类服务，以帮助开发人员加快新应用的开发和上市。DevOps 和容器可加快云原生应用的交付和部署，而应用服务则可加快云原生应用的开发。

例如，云原生应用开发人员可利用的应用服务，不仅能确保在基于容器的基础架构上顺利运行，也能充分利用各种平台功能，如 CI/CD 管道、滚动和蓝/绿部署、自动可扩展性、容错等。

步骤 4：欲善其事，必先择其利器

随着软件研究涉及的领域（物联网（IoT）、机器学习、人工智能（AI）、数据挖掘、图像识别、自动驾驶汽车等）越来越广泛，应运而生的软件开发框架、语言和方案也越来越多。

由于可根据特定的业务应用需求来选择语言或框架，而且这样的选择越来越多样，因此，云原生应用的构建方式也正变得越来越丰富。要应对因上述状况而不断增长的复杂性，您可以借助基于容器的应用平台，这类平台可以帮助您选择正确的框架、语言和架构组合来支持云原生开发。

要想实现云原生开发，还需要选用适当的工具来完成相应的任务。无论在实施云原生应用时采用的是 12 要素方案、基于域的设计、基于测试的设计和开发、MonolithFirst、快速单体式应用、迷你服务还是微服务，云原生平台都必须提供恰当的框架、语言和架构组合，以支持选定的开发需求。此外，基于容器的底层平台应该支持多种可随技术改进不断更新且受管辖的运行时和框架。

图 1. 云原生应用的开发变得越来越多样化

步骤 5：构建可按需提供的自助式基础架构

敏捷方法曾帮助开发人员快速创建和更新软件，但它欠缺有效的机制来确保开发人员能随时随地按需访问基础架构。在发布应用到生产环境时，整体上市速度就会受到影响。在基础架构价格低而工程人才成本高的时代，提交工单、等上数周，直到 IT 运维团队腾出资源已不再是一种可持续的工作模式。

通过自助式地按需置备基础架构，开发人员可以在需要时访问所需的基础架构。通过这种方式，可以有效消除未经授权的影子 IT。但是，只有当 IT 运维团队能够控制并了解不断变化且状况复杂的环境时，这种模式才有效。

容器和容器编排技术抽象并简化了对于底层基础架构的访问，并可针对各种基础架构环境（如数据中心、私有云和公共云）进行有效的应用生命周期管理。容器平台还可提供额外的自助服务、自动化和应用生命周期管理功能。该模式使开发人员和运维团队能够快速启用一致的环境，帮助开发人员专注于构建应用，而不会困于与置备基础架构相关的障碍和延迟。

标准化也是自助模式的要素之一。它有助于企业以一致的方式实现自动化和交付，达成业务目标。流程标准化会界定为执行某一任务（比如将某个应用发布到新环境）而需完成的事件和活动的具体顺序。

容器还有助于实现应用可移植性，包括创建可部署和运行于任何云提供商的云原生应用。实现可移植性后，可以在任意时间点自由地选择任意云提供商，轻松地从一个云提供商迁移到另一个云提供商，优化相关成本，并可在不针对特定云提供商 API 编写代码的情况下开发多云应用。

详细了解有助于开启云原生之旅的不同实践和技术。
探索开放实践库

步骤 6：实现 IT 自动化，加速应用交付

避免手动执行 IT 任务，是加速交付云原生应用的重点，而实现 IT 或基础架构自动化就是其中的关键所在。从网络和基础架构置备，到应用部署和配置管理，自动化可以整合并应用于任何任务或组件。

IT 管理和自动化工具会创建可重复的流程、规则和框架，以替代或减少会延迟上市的劳动密集型人工介入。 这些工具可以进一步延伸到具体的技术（如**容器**）、方法（如 **DevOps**），再到更广泛的领域（如**云计算**、安全性、测试、监控和警报）。因此，自动化是 IT 优化和数字化转型的关键，可以缩短实现价值所需的总时长。

IT 自动化指南

1. 请采用企业级编程自动化方案来开展 IT 运维。在整个企业内开展协作对话，以定义各种服务需求。
2. 考虑利用自动化沙盒，为学习自动化语言和流程奠定基础。
3. 认真思考自动化。确保消除任何不必要的手动步骤，即使该手动操作足够让您高枕无忧。
4. 考虑采用系统化方法，每次推进一小步，逐步实现自动化。每个步骤间环环相扣，从而实现覆盖广泛的自动化实践。
5. 可以优先实现某一项任务或服务（计算、网络、存储或置备）的自动化。与他人分享这一自动化成果，并基于此进行系统化构建。
6. 利用自助服务目录，为用户授权并加速交付。
7. 对各种策略和流程进行计量、监控和退款。

随着时间的推移，您不但能实现全面的集成式自动化，还能提高效率、加快 DevOps 速度并实现快速创新。

详细了解 IT 自动化在“自动化企业”中扮演的重要角色
下载电子书

持续交付（CD）是一种软件工程方案，它能让团队以较短的周期持续提供有价值的软件，同时确保软件随时都能可靠地发布。借助可靠的低风险版本，CD 可以不断地调整软件，以响应用户的反馈，并应对不断变化的市场和业务战略。

Gartner 的定义

步骤 7：实施持续交付和高级部署技术

发布周期较长，不仅意味着发现软件错误到解决之间所耽误的时间较长，还如同一道天生障碍，让您难以及时响应客户和市场需求。对于高流量应用（如移动、Web 或物联网应用），未解决的错误可能会影响许多用户，从而导致客户体验欠佳、安全或防护问题、生产力或收入下降。即使对于其他内部业务应用，因解决软件错误而发生的中断或延迟也会产生高昂的业务成本。

敏捷开发方法经过不断演变，形成了“早发布，常发布”模式。DevOps 和持续交付方案通过密切联合开发人员、运维、质量保证和安全团队，扩展了这些方法的应用领域，从而改善了软件的交付流程。因此，代码的变动可以快速可靠地推送至生产环境，为开发人员提供快速反馈。这种迭代式快速反馈循环借助 CI/CD 实现， 可将基础架构自动化扩展到端到端自动交付系统，从而涵盖应用交付的方方面面，包括自动化测试、漏洞扫描、安全合规性和法规检查。自动化交付管道旨在不影响运维能力的情况下提供更新，同时降低交付风险。

要实现持续交付（CD），首先要实现持续集成（CI）。CI 系统是一组 Build 系统，它们可以监控各种源代码控制存储库的变更情况，运行任何适用的测试，并通过每个源代码控制变更自动构建最新的应用版本，如 Jenkins。

查看红帽 Ansible® 自动化等现代自动化技术如何支持 CI/CD
下载白皮书

“先进的部署技术能在结构和透明度方面实现创新。成熟的部署方法能够构建一个真正实现实验、反馈和分析的环境。更好的实验有助于实现更好的创新。”

BURR SUTTER
红帽开发人员体验总监
REDHAT.COM/ZH/ENGAGE/TEACHING-AN-ELEPHANT-TO-DANCE

高级部署模式旨在降低与软件发布相关的风险，构建结果受控且不会对客户造成意外负面影响的实验环境。该目标对于推动整个企业不断创新至关重要。

高级部署技术将交付的性质从非工作时间周末活动（有服务期限和停机时间）变为日常工作日活动（生产中不存在停机时间），同时确保应用仍可供客户使用。

通过消除新部署对客户造成的不便，这些技术使企业能够按照业务所需的频率来交付更新和发布版本。以下是一些可用于实现零停机部署的常用部署技术，具体技术根据用例而定：

滚动部署模式不会一次更新应用的所有实例，它会依次将各个实例从负载均衡器中排除，使其无法接收流量，然后再单独更新各个实例。实例更新后，会再次纳入负载均衡器中。此流程会持续进行，直到所有实例都已更新。

蓝/绿部署描述的是运行两个相同环境的做法：一个环境处于活动状态，而另一个处于空闲状态。所做的更改会被推广至空闲环境；然后，待更改在生产中完成验证后，实时流量会切换到已更新的环境。回滚到先前版本就像切换回流量一样简单，但前提是得考虑数据的传输。

Canary 部署类似于蓝/绿部署，需使用两个相同的环境。但是，它的不同之处在于发布的控制方式。部署新版本后，会向一小部分客户发送新版本，供其在生产中进行测试。如新版本验证成功，则流量会逐渐转移到新版本，同时监控并验证结果，直到向所有用户发送完新版本。

教大象跳舞
下载电子书

步骤 8：推动变革，采用模块化程度更高的架构

在基于微服务的软件编写架构方案中，应用被拆分成最小的组件，并彼此独立。不同于将所有组件内置于同一架构中的传统单体式方案，微服务都是独立的组件，通过合作来完成相同的任务。此种软件开发方案强调高精度、轻量化，力求在多个应用中共享相似的流程。虽然微服务架构不要求使用特定的底层基础架构，但基于容器的平台可以打下最扎实的基础。

通过持续变革基于微服务的架构，有望为超大型团队提供更多优势，还可以在一天内多次执行生产部署。从架构的角度来看，微服务需将每一个服务拆分成各自的部署单元。随后，用户可单独管理和部署每个微服务，并且可能由不同的团队来负责各个微服务的生命周期。

但是，实施微服务架构需要一定的投资和技能，可能会对企业造成过大的负面影响。分析师和主题专家建议对微服务采用 MonolithFirst 方案，即要先构建一个单体式应用，就算您想创建的是微服务架构。这么做的目的是：先充分理解您的应用所属的域，然后更好地识别其所含的有限上下文——这些上下文将作为转换成微服务的候选内容。这样做，有助于避免技术债务，比如还没有了解应用的所属域和有限上下文就构建一组微服务，由此产生的修复成本。

微服务的另一种替代方案是迷你服务。迷你服务是按域划分的多个服务的集合，通常在应用服务器上运行。迷你服务可以提高敏捷性并进行扩展，而无需担心基于微服务的设计和基础架构所带来的复杂性问题。迷你服务也需要针对敏捷性、DevOps 和 CI/CD 方案进行投资，确保现代化应用服务器或多框架、多架构和 多语言产品与基于容器的基础架构完美结合。

能够支持不同的框架、语言和云原生应用开发方案的平台（例如，微服务、迷你服务或 MonolithFirst）是云原生应用成功的关键。

6. 云原生应用业务案例

在实现数字化转型时，各个企业都有不同的优先事务。有些企业会遵循现代化的云原生原则，使其现有的应用架构和基础架构实现现代化转型，而有些企业会利用全新的业务模式和应用来实现创新。无论具体的意图和业务用例如何，这些企业的目标都是提高速度和灵活性，并为数字化转型做好准备。云原生应用的常见用例大致可根据以下四项业务挑战进行分类：

*“在收购了一家新银行的**第 1 天**，我们在生产环境中应用了 **10 项变更**，**没有出现任何问题**。”*

JOHN RZESZOTARSKI
KEYBANK DEVOPS 总监

部署时间缩短：
从 12 周缩短为 1 周

截至 2023 年，90% 的现有应用仍会继续使用，但其中的大部分应用都无法获得足够的现代化改造资金支持。

Gartner：“Application Modernization should be business-centric, continuous, and multiplatform”（应用的现代化应以业务为中心持续开展，并实现多平台适用性），2018 年 1 月。

“红帽会为 JBoss 企业应用平台提供专家支持，所以我们不必为日常运营而担心。”

企业技术部
前任总经理，
澳大利亚证券交易所

应用重启速度**提高了 60 倍**

平台支持所耗成本和时间均有**减少**，为实现创新服务开发腾出资源

业务挑战 1：加速应用交付

目标：
加快向客户交付现有应用和新应用的速度。

方案：
容器可以提供一个通用平台，使开发、运维、安全性、质量保证和其他团队联合起来，一同采用独立于基础架构和应用技术的 DevOps。借助 DevOps 方案，团队可以采用自动化及 CI/CD 实践来快速、放心地发布软件。利用基于容器的自动化功能解决部署问题后，可以缩短应用的交付周期，并使其适应业务的发展步伐，而不是根据 IT 部门所能交付的内容来调整业务的发展步伐。

KeyBank

客户聚焦：
KeyBank 是美国排名前 15 的银行之一。该公司推出了一项数字渠道现代化计划，旨在更新其网络体验并创建一个全新的移动网络应用。在红帽 OpenShift® 的帮助下，KeyBank 从单体式应用迁移到了微服务，从而构建了一个自动化的持续交付管道，并将部署频率从每个季度一次改成了每周一次。

业务挑战 2：对现有应用进行现代化改造

目标：
对现有应用进行现代化改造，以满足市场和客户需求，从而加速实现变革。

方案：
许多产生价值的业务应用都是传统应用，其设计年代可能远早于数字时代。但是，直接淘汰换新不一定就是最优对策，至少在经济层面上并不可行。此外，也并非所有的传统应用都能现代化。

如果将传统应用迁移至云环境行得通，容器就可以为其提供相应的支持，消除对于底层基础架构的依赖性。这样，应用就能从企业内部基础架构移植到云端；如果需要，还可以对应用进行重构和重新架构，以实现云原生。容器平台方案还可利用平台的自动化功能和 DevOps 实践来简化现有应用的迁移。

ASX

客户聚焦：
作为每天第一个开市的主要金融市场，澳大利亚证券交易所（ASX）在全球金融服务领域占据重要地位。该机构必须确保高稳定性、高安全性和高性能运营，但其传统应用服务器平台的失常和不稳定程度正在加剧，相关的成本也在日益增长。ASX 制定了一项计划，旨在借助全新技术实现其数字平台的现代化；该机构还决定部署红帽 JBoss® 企业应用平台，以便为其应用服务器奠定坚实的基础。最开始的部署涉及到该机构一个重要的 B2B 网页应用“ASX Online”，其主要用于向市场提供价格、公司公告和关键报告，同时满足相关的法规要求。

“我已经彻底爱上了红帽 OpenShift 容器平台。这款创新型产品能让我们快速完成部署并轻松实现容器控制。”

MICHAEL AALBERS
高级技术应用协调员，
阿姆斯特丹史基浦机场

*新 API 的开发速度提高了 **50%***

业务挑战 3：开发新的云原生应用

目标：
加快新应用的开发速度，把握各种新商机。

方案：
如果企业能将理念快速转化为服务和产品、快速评估它们在新领域中所取得的业务成果，然后再进行快速的调整，那么不断变化的业务和客户需求将为企业带来巨大的商机。集基于服务的架构、API 整合、容器化服务和编排以及各种 DevOps 实践、自动化功能和工具的强大支持于一身，旨在构建新应用的云原生方案可以加速从理念到创新应用的转化过程。

Schiphol
Amsterdam Airport

客户聚焦：
史基浦国际机场是欧洲第三大繁忙的机场，每年要接待旅客 6400 万人次。该机场的目标，是在 2018 年前转型成为全球一流的数字化机场。为了实现这一目标，它需要借助云无关的平台来加速应用的开发。史基浦数字化战略的要点之一，是通过 API（包括“航班”API）提供服务，航班 API 可为旅客提供登机口、候机楼和登机手续办理时间方面的信息。借助红帽 OpenShift 容器平台，史基浦将为其内部 IT 团队和业务伙伴创建一个自助式多云平台，从而缩短新服务的开发时间。

“在整个过程中，最令人激动的是我们将重新设计 IT 部门的工作方式。我们将彻底改变作为一个企业所采用的工作方式，[并]开始着手改变整个银行的工作方式。”

WAYNE MARCHANT
HERITAGE BANK 首席信息官

业务挑战 4：推动业务创新

目标：
加快整个企业的创新速度，以满足不断变化的业务需求。

方案：
面对日新月异的世界，停滞不前就意味着落后。IT 团队正在你追我赶，力图快速推出各种新功能和服务，让客户满意并帮助员工更智能地工作。成功取决于不断创新，而不仅仅依赖于新的工具和技术。要想取得成功，需要推行变革性的新文化、新工具和新流程，以便于在整个企业内实现创新和开展实验。

Heritage Bank

客户聚焦：
Heritage Bank 已创立 142 年，是澳大利亚历史最悠久的金融机构之一。面对日益激烈的市场竞争和新的市场要求，Heritage Bank 需要寻求新的方式来更快速地交付软件。通过红帽开放创新实验室的团队拟真协作，Heritage Bank 打造了一款创新的银行解决方案，并组建了一支绩效卓越的团队。该团队现在能够持续快速开发出更好的软件，从容应变未来的各种需求。

Heritage Bank 的视频

红帽可以为您提供帮助

针对您在数字化和云原生之旅中所处的阶段以及您优先考虑的事宜，红帽的技术和服务可助您一臂之力。

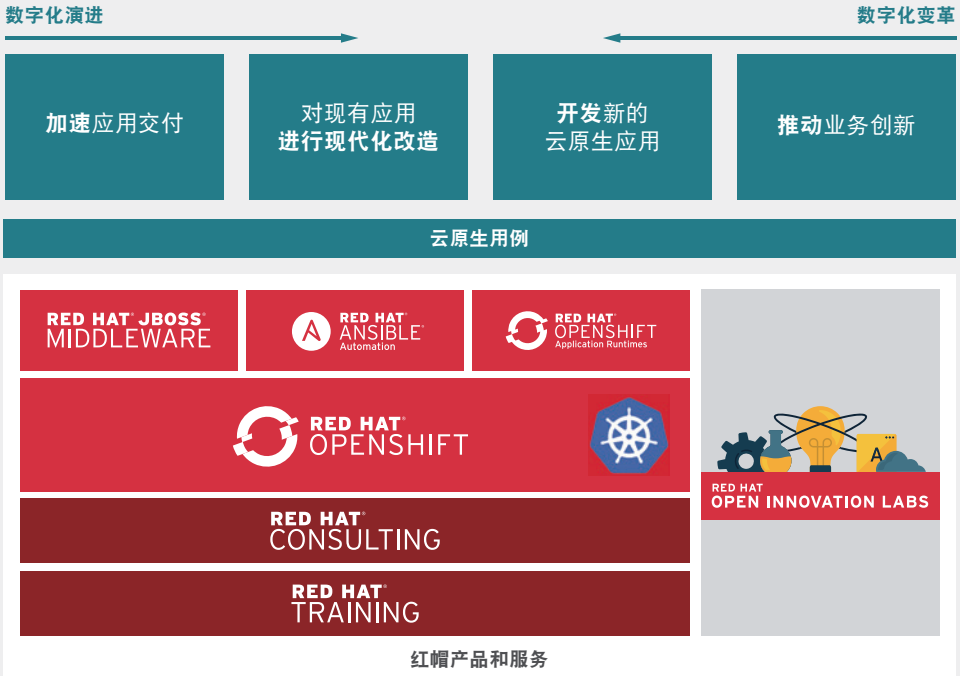


图 2. 借助红帽开启云原生之旅

一些企业可能只会重点关注一个云原生用例，而另一些企业可能会同时优先考虑多个用例。无论您采用的是渐进演变式还是剧变革命性方案，您的云原生之旅都会高度个性化，而且不一定是线性模式。无论您选择哪种路径，要想加快应用的上市速度，都需要推行适当的技术、DevOps 实践和文化。

红帽可以借助云原生容器开发平台——**红帽 OpenShift** 帮助您完成这一旅程。**红帽 OpenShift 应用运行时**可以提供开源运行时和框架，用于通过 OpenShift 上的容器化运行时构建云原生应用，并缩短开发时间。您可以在 OpenShift 上部署各种红帽 JBoss 中间件技术，包括 Ansible **自动化和管理技术**。

为帮助您应对数字化转型的复杂性，**红帽咨询**可以提供相关的战略性建议，以及深入的技术专业知识。从**红帽开放创新实验室**到业务探讨和项目实施计划，我们的顾问将协助您走过云原生之旅中的每一步。

您是否已开启云原生之旅?

详细了解红帽如何协助您完成云原生应用的构建之旅:

- 了解红帽咨询可以提供哪些帮助: 通过提供咨询的业务探讨, 获得最佳实践和规划指导。
- 查看我们的服务之声博客, 获取相关的洞见信息、技术窍门等。
- 您的 DevOps 成熟度如何? 您已为云原生之旅做好了哪些准备? 完成 Ready To Innovate 评估, 明确自己的准备情况。

关于红帽

红帽是世界领先的开源解决方案供应商, 依托社区力量为客户提供稳定可靠及高性能的云技术、Linux、中间件、存储和虚拟化产品。红帽还提供屡获殊荣的支持、培训和咨询服务。作为紧密连接全球企业、合作伙伴和开源社区的中心, 红帽致力于通过为广大客户提供实用、创新型技术产品, 有效释放其宝贵资源以推动业务增长, 并为未来 IT 发展奠定坚实基础。

*查看更多**红帽产品组合**信息, **请访问 redhat.com/zh***

销售及技术支持

800 810 2100
400 890 2100

红帽软件 (北京) 有限公司

北京市朝阳区东大桥路 9 号侨福芳草地大厦 A 座 8 层　邮编: 100020
86 10 6533 9300

红帽官方微博　红帽官方微信

cn.redhat.com
#F12255_0518

版权所有 © 2018 Red Hat, Inc。红帽、红帽企业 Linux、Shadowman 徽标、OpenShift、Ansible 和 JBoss 是 Red Hat, Inc. 在美国和其他国家/地区的商标或注册商标。Linux® 是 Linus Torvalds 在美国和其他国家/地区的注册商标。